Tampereen yliopisto

Petri Erola

OHJELMISTO-OHJATTUJEN TIETOVERKKOJEN PROTOKOLLAT

Informaatioteknologian ja viestinnän tiedekunta
Kandidaatintyö
Elokuu 2019

TIIVISTELMÄ

Petri Erola: Ohjelmisto-ohjattujen tietoverkkojen protokollat
Kandidaatintyö
Tampereen yliopisto
Tieto- ja sähkötekniikan kandidaatin tutkinto-ohjelma
Elokuu 2019

Tämä työ käsittelee ohjelmisto-ohjattujen tietoverkkojen protokollia. Työssä käydään läpi tietoverkkojen rakenteita ja erityisesti perehdytään ohjelmisto-ohjattujen tietoverkkojen arkkitehtuuriin. Työssä esitellään mihin käytännössä ohjelmisto-ohjattuja tietoverkkoja voidaan soveltaa. Tarkemmin työssä käsitellään ohjelmisto-ohjatuissa tietoverkoissa käytettäviä protokollia ja tarkastellaan niiden ominaisuuksia. Tämä tutkimus on tehty kirjallisuusselvityksenä.

Ohjelmisto-ohjatut tietoverkot eli Software-Defined Networking on ajatusmalli, jossa perinteisen tietoverkon datataso ja hallintataso on erotettu toisistaan. Näiden tasojen erottamisen seurauksena tietoverkkoa on mahdollista tarkastella abstraktiona ja sitä voidaan käsitellä ohjelmallisesti. Tietoverkkojen ohjelmoitavuus nopeuttaa ja helpottaa niiden hallintaa. Helpottuneen hallinnan lisäksi ohjelmoitavien verkkojen ja uusien tietoliikenneprotokollien kehittäminen nopeutuu, koska verkon hallinta ei riipu enää laitteistovalmistajista.

Ohjelmisto-ohjattavan tietoverkon toteuttaminen vaatii tapaa viestiä hallintatasolta datatasolle. Tätä viestintää voidaan toteuttaa erilaisilla protokollilla. Protokollat voidaan jakaa niiden tehtävien mukaan siten, että datatason datapakettien välityksen määrittävät protokollat ovat omassa ryhmässään ja datatason laitteistoja konfiguroivat protokollat ovat omassa ryhmässään.

Avainsanat: Software-Defined Networking, SDN, OpenFlow, tietoverkko, protokolla

SISÄLLYSLUETTELO

LYHENTEET JA MERKINNÄT

API	Application Programming Interface
AS	Autonomous System
BGP	Border Gateway Protocol
BGP-LS	Border Gateway Protocol Link State
CE	Control Element
D–CPI	Data–Control Plane Interface
FE	Forwarding Element
ForCES	Forwarding and Control Element Separation
HTTP	Hyper Text Transfer Protocol
IAB	Internet Architecture Board
IETF	Internet Engineering Task Force
IP	Internet Protocol
JSON	JavaScript Object Notation
LFB	Logical Functional Block
LLDP	Link Layer Discovery Protocol
LSP	Label Switched Path
MAC	Media Access Control
MI	Management Interface
NBI	Northbound Interface
NE	Network Element
NETCONF	Network Configuration Protocol
NFV	Network Functions Virtualization
OF-Config	OpenFlow Management and Configuration Protocol
ONF	Open Networking Foundation
OSPF	Open Shortest Path First
OVS	Open vSwitch
OVSDB	Open vSwitch Database
PCC	Path Computation Client
PCE	Path Computation Element
PCEP	Path Computation Element Protocol
QoS	Quality of Service
RIP	Routing Information Protocol

RPC	Remote Procedure Call
SBI	Southbound Interface
SDN	Software-Defined Networking
SDRAM	Synchronous Dynamic Random Access Memory
TCP	Transmission Control Protocol
TED	Traffic Engineering Database
TLS	Transport Layer Security
XML	Extensible Markup Language

1. JOHDANTO

Mobiililaitteiden määrä sekä konesaliverkkojen kasvu ovat aiheuttaneet tarpeen tietoverkkojen hallinnan automatisoinnille. Nykyajan yritysverkot ovat monimutkaisia ja sisältävät erilaisia verkkolaitteita [1]. Varsinkin suuret tietoverkot sisältävät suuren määrän verkkolaitteita, joiden avulla verkkoa hallitaan. Perinteisissä tietoverkoissa verkkolaitteita hallitaan paikallisesti laite kerrallaan. Näin ollen verkkojen muokkaus on hidasta ja kankeaa. Tämän takia tietoverkkojen hallinnan automatisointi on ollut viime vuosina tärkeä tutkimuskohde. Perinteisten tietoverkkojen korvaajaksi on ehdotettu ohjelmisto-ohjattuja tietoverkkoja. Ohjelmisto-ohjattu tietoverkko on ajatusmalli, jossa verkon datapakettien lähettäminen ja verkon hallinta on erotettu toisistaan. Tämä erottaminen mahdollistaa verkon hallinnan erillään datapaketteja siirtävästä verkkolaitteesta, mikä tarkoittaa sitä, että kaikkien verkkolaitteiden hallinta voidaan keskittää samaan paikkaan. Verkon hallinta ja muokkaaminen helpottuu huomattavasti, kun se voidaan tehdä yhdessä, erillisessä paikassa. Nykyajan tietoverkoissa kulkee myös paljon erilaista dataa. Resurssien jakamista erilaisen datan kesken kutsutaan palvelun laaduksi. Palvelun laadun joustava ja edullinen hallinta ei ole perinteisissä tietoverkoissa mahdollista. Ohjelmisto-ohjattavan tietoverkon uskotaan ratkaisevan tämänkin ongelman.

Tässä työssä tutkitaan ohjelmisto-ohjattavia tietoverkkoja. Erityisesti tutustutaan ohjelmisto-ohjattavien tietoverkkojen protokolliin. Tutkittava aihe on rajattu koskemaan täysin ohjelmallisesti ohjattavia tietoverkkoja ja niin sanottuja hybridiverkkoja eli perinteisten ja ohjelmisto-ohjattavien verkkojen yhdistelmiä ei käsitellä erikseen tässä työssä. Työssä pyritään selvittämään, mitä protokollia ohjelmisto-ohjatut tietoverkot käyttävät, ja miten eri protokollat vertautuvat toisiinsa. Tutkimus on tehty perehtymällä aiheesta löytyvään kirjallisuuteen.

Toisessa luvussa tarkastellaan aluksi, miten perinteiset tietoverkot toimivat. Sen jälkeen perehdytään ohjelmisto-ohjattuun tietoverkkoon ja sen arkkitehtuuriin. Toisen luvun lopuksi käydään läpi esimerkkejä ohjelmisto-ohjattujen tietoverkkojen käyttökohteista. Kolmannessa luvussa käsitellään ohjelmisto-ohjatuissa tietoverkoissa käytettäviä protokollia. Protokollat on jaoteltu tehtäviensä mukaan omiksi ryhmiksi. Tässä osassa tarkastellaan protokollien ominaisuuksia ja verrataan keskenään vaihtoehtoisia

protokollia toisiinsa. Yhteenvedossa kootaan työn tulokset ja tehdään johtopäätökset ohjelmisto-ohjatuista tietoverkoista ja niissä käytetyistä protokollista.

2. TIETOVERKKORAKENTEET

2.1 Perinteiset tietoverkot

Perinteinen tietoverkko muodostuu verkossa olevista laitteista ja niiden välisistä yhteyksistä. Tietoverkot koostuvat päätelaitteista, kuten tietokoneista ja palvelimista, erilaisista kytkimistä, reitittimistä, palomuureista ja kuormantasaajista. Laitteet voivat olla yhteydessä toisiinsa langallisesti kaapeleilla tai langattomasti radiotien välityksellä. Tietoverkon tarkoitus on luoda yhteyksiä päätelaitteiden välille. Kytkimiä ja reitittimiä tarvitaan yhdistämään useita laitteita toisiinsa.

Tietoverkoissa siirrettävä data on nykyään pakettimuotoista eli verkot ovat pakettikytkentäisiä. Pakettikytkentäisessä verkossa lähetettävät paketit ohjataan niiden otsikossa olevien tietojen perusteella eteenpäin. Datapaketin otsikkokentässä on tieto paketin kohdeosoitteesta, jonka perusteella verkkolaite osaa lähettää paketin eteenpäin [2]. OSI-mallin (Open Systems Interconnection Reference Model) verkkokerroksella tapahtuva paketin lähettäminen kohdeosoitteeseen tapahtuu paketin otsikosta löytyvän IP-osoitteen (Internet Protocol) perusteella. IP-osoite yksilöi kohteen verkkotasolla. Verkkotasolla toimivat reitittimet käyttävät reititykseen reititystauluja. Reititystaulussa sijaitsee tiedot reitittimeen yhteydessä olevien laitteiden osoitteista. Reititin etsii saapuvan paketin kohdeosoitteen reititystaulusta ja kytkee paketin sen mukaan eteenpäin. [2] Jokainen reititin, johon paketti matkallaan saapuu, tarkastelee pakettia erikseen ja kytkee paketin oman reititystaulunsa perusteella. OSI-mallin siirtokerroksella tapahtuva pakettien lähettäminen tapahtuu MAC-osoitteiden perusteella. MAC-osoite (Media Access Control) on laitteen siirtokerrostasolla yksilöivä tunnus, jonka mukaan kytkimet siirtävät paketteja.

Perinteisissä verkoissa reititys voidaan jakaa staattiseen tai dynaamiseen reititykseen. Staattisessa reitityksessä reitit määritellään manuaalisesti ja ne pysyvät samoina koko ajan. Dynaamisessa reitityksessä reitittimet kommunikoivat toisilleen ja jakavat tietoja omista yhteyksistään reititysprotokollien välityksellä. Tällä tavalla kaikki reitittimet tulevat tietoisiksi kaikista verkoista. Staattiselle ja dynaamiselle reititykselle yhteistä on se, että reititystapa täytyy manuaalisesti asettaa reitittimelle. Usein käytetään staattisen ja dynaamisen reitityksen yhdistelmää, jossa harvoin muuttuvan verkon osan reititys määritetään staattisesti ja usein muuttuvan osan reititys dynaamisesti. Staattisen reitityksen aiheuttama yleisrasite (engl. *overhead*) on pienempi kuin dynaamisen

reitityksen, eli se säästää kanavan resursseja. Toisaalta, staattisen reitityksen määrittäminen suuressa verkossa on haastavampaa kuin dynaamisen reitityksen. [2] Reititysprotokollia ovat esimerkiksi RIP (Routing Information Protocol) [3], OSPF (Open Shortest Path First) [4] ja BGP (Border Gateway Protocol) [5]. RIP ja OSPF laskevat etäisyyksiä verkon solmupisteiden välillä ja valitsevat lyhyimmän reitin. RIP laskee verkkojen välillä olevien hyppyjen määrän, kun taas OSPF laskee reittien pituudet niiden yhteysnopeuksien perusteella. BGP määrittää reitityksen autonomisten järjestelmien (autonomous system, AS) välillä. AS:t ovat suuria verkkokokonaisuuksia, kuten verkko-operaattoreiden kokonaisverkkoja.

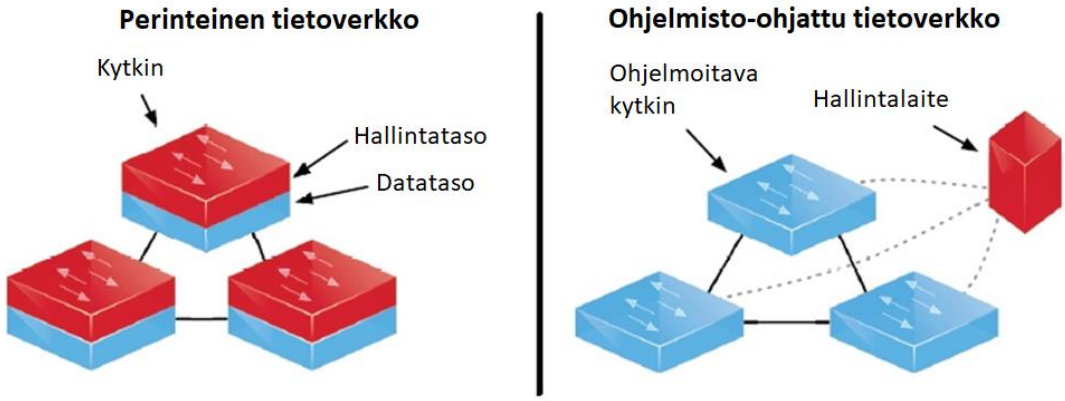
Perinteisten tietoverkkojen konfigurointi on hidasta ja monimutkaista [6]. Jos pakettien reititykseen ja lähettämiseen liittyviin sääntöihin halutaan tehdä muutoksia, täytyy muutokset tehdä jokaiselle paketteja välittävälle verkkolaitteelle erikseen. Varsinkin suurissa verkoissa se on ongelma. Jäykkien verkkojärjestelmien takia uuden reititysprotokollan kehittäminen ja käyttöönotto voi kestää vuosia [6]. Palvelun laadun (engl. *Quality of Service*, QoS) hallinta ei onnistu perinteisissä verkoissa joustavasti [7]. Modernien verkkojen tarpeisiin kuuluu monenlaisten eri yhteyksien tukeminen, joten QoS:n ketterä muokkaus on tarpeellista.

2.2 Ohjelmisto-ohjatut tietoverkot (SDN)

Ohjelmisto-ohjattu tietoverkko (Software-defined Networking, SDN) on ajatusmalli, jossa verkon hallinta on erotettu pakettien välityksestä. Perinteisissä tietoverkoissa kytkimet ja reitittimet vastaavat sekä reititykseen liittyvästä laskennasta että pakettien lähettämisestä eteenpäin. SDN:ssä pakettien välittäminen tapahtuu omalla tasollaan ja verkon hallinta omalla tasollaan. Kreutz et al. [6] määrittelevät SDN:n neljän periaatteen mukaan: datatason ja hallintatason erottaminen toisistaan, pakettien välittäminen vuohon perustuen, verkon hallinnan siirtäminen erilliselle ohjainyksikölle sekä verkon ohjelmoitavuus. Hoang ja Pham [8] pitävät myös avoimuutta tärkeänä osana SDN:ää.

Perinteisissä tietoverkoissa pakettien välittäminen ja verkon hallinta on integroituna samaan laitteeseen. SDN:ssä datapakettien välittäminen ja verkon hallinta on erotettu omiksi tasoiksi. Pakettien välittämisestä vastaavat datatason verkkolaitteet eli SDN-kytkimet, ja verkon hallinta kuuluu erilliselle verkon ohjaimelle, joka on loogisesti keskitetty. Verkon hallinnan erottaminen pakettien lähettämisestä helpottaa verkon muokkaamista ja kehittämistä, sillä keskitetyn ohjaimen kautta ohjelmoidusti tehdyt

muutokset saadaan käyttöön koko verkkoon samaan aikaan. [9] Perinteisissä verkoissa muutokset täytyy tehdä jokaiselle verkkolaitteelle erikseen. Tasojen erottaminen toisistaan helpottaa verkon hallintaan tarkoitettujen toimintojen kehittämistä, koska ne eivät enää riipu valmistajien laitteista. SDN mahdollistaa verkon ohjelmoitavuuden, jolloin verkon konfigurointi on perinteistä reititinmallista verkkoa joustavampaa ja nopeampaa. Kuvassa 1 on havainnollistettu data- ja hallintatason erottamista SDN:ssä. Keskitetyn hallinnan haasteena on sen turvallisuus. Jos hyökkääjä pääsee käsiksi erilliseen verkon ohjaimeen, voi hyökkääjä päästä vaikuttamaan koko verkkoon. Lisäksi skaalautuvuus on haastavaa, koska yksi ohjain voi hallita vain rajallisen määrän SDN-kytkimiä.



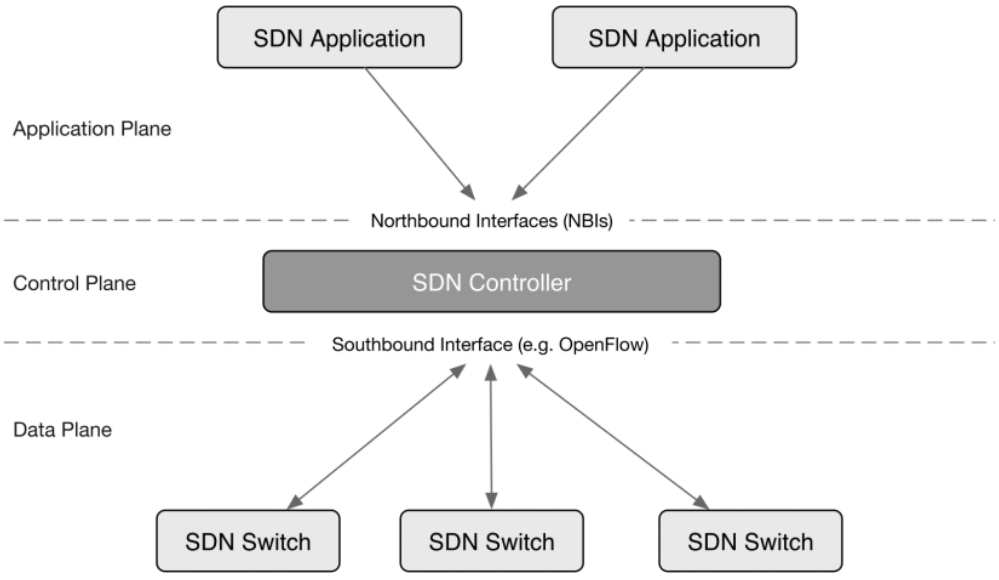
Kuva 1. Data- ja hallintatasot perinteisissä sekä ohjelmisto-ohjatuissa verkoissa. Muokattu lähteestä [10]

Tietoliikennekanavassa datan lähteen ja kohteen välistä pakettivirtaa kutsutaan vuoksi (engl. *flow*). SDN:ssä pakettien välittäminen perustuu pakettivuon informaatioon, toisin kuin perinteisissä verkoissa, joissa paketit välitetään pelkästään osoitteiden perusteella. Vuon informaatiota voivat olla esimerkiksi paketin lähde- ja kohdeosoitteet, VLAN-tunnukset tai paketin käyttämä kuljetusprotokolla. Pakettien välittäminen vuon perusteella mahdollistaa palvelun laadun paremman tuottamisen [11]. Tämän taustalla on se, että paketit voidaan välittää niiden ominaisuuksien perusteella sen sijaan että ne välitettäisiin pelkästään niiden osoitteiden perusteella. Samassa kanavassa voidaan siis siirtää eri tason dataa, esimerkiksi HD-laatuista suoratoistoa ja tavallisia HTTP-pyyntöjä (Hypertext Transfer Protocol) ilman, että HD-kuvanlaatu kärsii tai yksinkertaisiin HTTP-pyyntöihin kulutetaan liikaa resursseja. SDN:n ohjelmoitavuus mahdollistaa myös palvelun laadun ketterän muokkauksen.

SDN:ssä eri tasojen väliset rajapinnat ovat avoimia, joten SDN on vähemmän valmistajariippuvainen kuin perinteiset tietoverkot. Avoimet rajapinnat sekä data- ja hallintatason erottaminen helpottavat tietoverkkotekniikoiden kehittämistä ja innovointia [12]. Aiemmin tietoverkkojen kehitystyö oli laitteistojen valmistajista riippuvaista mutta SDN:n myötä verkon hallintaan liittyviä toimintoja voidaan kehittää erillään laitteista. Avoimet rajapinnat mahdollistavat sovelluksien vapaan kehittämisen. SDN-verkkojen kehitystä varten perustettu Open Networking Foundation (ONF) mainitsee strategiassaan yhdeksi periaatteekseen avoimen lähdekoodin alustojen luomisen [13]. Avoimista rajapinnoista huolimatta SDN:ää ei voida toteuttaa perinteisillä verkkolaitteilla. Verkkoinfrastruktuuri on siis uusittava SDN-yhteensopivilla verkkolaitteilla, mikäli haluaa toteuttaa täysin ohjelmoitavan verkon.

2.3 SDN-arkkitehtuuri

SDN:ää voidaan mallintaa jakamalla se kolmeen osaan: datatasoon (engl. *data plane*), hallintatasoon (engl. *control plane*) ja sovellustasoon (engl. *application plane*). Hallintataso voidaan jakaa vielä kahteen osaan (control plane ja management plane) mutta tässä työssä hallintatasolla viitataan näihin molempiin. Eri tasot ovat yhteydessä ylä- ja alapuolella oleviin tasoihin erityisten rajapintojen välityksellä. Hallintatason ja datatason välistä rajapintaa kutsutaan Southbound interface:ksi (SBI). SBI:n välityksellä hallintatason SDN-ohjain ohjaa datatason SDN-kytkimiä. Hallintatason ja sovellustason välistä rajapintaa kutsutaan Northbound interface:ksi (NBI). Sovellustasolta tulevat käskyt verkon hallintaan välitetään ohjaimelle NBI:n kautta.



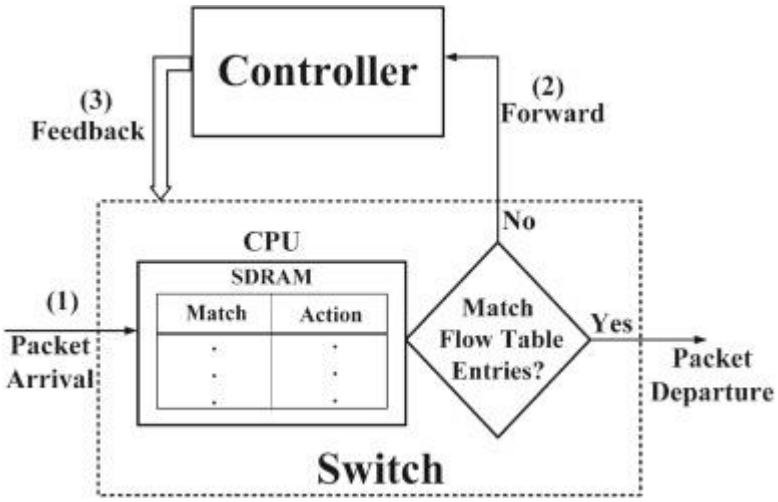
Kuva 2. SDN-tasot. [14]

Kuvassa 2 on esitetty SDN-arkkitehtuurin eri tasot ja rajapinnat. Lisäksi kuvassa 2 on eri tasojen tärkeimmät komponentit. Datatasolla sijaitsee SDN-kytkimet (SDN Switch), hallintatasolla SDN-ohjain (SDN Controller) ja sovellustasolla verkon hallintaan tarkoitetut SDN-sovellukset (SDN Application). Tässä luvussa tarkastellaan SDN:n kolmea tasoa sekä kerrotaan niiden toiminnasta ja rakenteista.

2.3.1 Datataso

Datatasolla tapahtuu pakettien välittäminen eteenpäin. Datataso koostuu verkkolaitteista, joita yleensä kutsutaan SDN-kytkimiksi. SDN-kytkimet välittävät paketteja eteenpäin verkossa, ja ne korvaavat perinteisen tietoverkon kytkimet ja reitittimet. SDN-kytkimet saavat ylemmältä tasolta ohjeet, kuinka paketteja tulee kytkeä. Käytännössä nämä ohjeet tarkoittavat vuotauluja (engl. *flow table*). Vuotaulu on kooste säännöistä, joita SDN-kytkin noudattaa lähettäessään pakettia eteenpäin [15].

SDN-kytkin voi olla joko ohjelmistokytkin (engl. *software switch*) tai perinteinen laitteistopohjainen kytkin (engl. *hardware switch*). Kuvassa 3 on yleinen lohkokaavio SDN:ssä käytetystä ohjelmistokytkimestä.



Kuva 3. Yleiskuvaus ohjelmistokytkimestä. [16]

Kuvasta 3 nähdään, että SDN-kytkimellä on sisäänmenolle (Packet Arrival) ja ulostulolle (Packet Departure) portit, joiden kautta paketit kulkevat kytkimen läpi. SDN-kytkin voi sisältää yhden tai useamman vuotaulun. Vuotaulut säilytetään SDRAM-muistissa (Synchronous Dynamic Random Access Memory). SDN-kytkimessä on paketin prosessointia varten suoritin, joka vertaa saapunutta pakettia muistissa olevaan vuotauluun. Jos vuotauluja on useampi, pakettia verrataan järjestyksessä kaikkiin

vuotauluihin, kunnes vastaavuus löytyy tai kaikki taulut on käyty läpi. Jos SDN-kytkin ei osaa yhdistää sille tulevaa pakettia vuotaulun merkintöihin, paketti käsitellään ennalta määrätyllä tavalla. Paketti voidaan esimerkiksi pudottaa, lähettää kaikkiin portteihin tai lähettää ylemmän tason SDN-ohjaimelle, joka palauttaa paketin välitykseen tarvittavan informaation ja päivittää SDN-kytkimen vuotaulun [16]. Päivitetyn vuotaulun avulla paketti osataan lähettää eteenpäin. SDN-kytkin ei itse osallistu reitin määritykseen tai hallintaan, vaan se ainoastaan vertaa pakettia tunnettuihin sääntöihin ja lähettää paketin sen mukaan eteenpäin.

2.3.2 Hallintataso

Hallintatasolle kuuluvat verkon hallintaan liittyvät toiminnallisuudet, kuten verkkotopologian löytäminen, vuotaulujen päivittäminen SDN-kytkimille, verkon muutoksien hallinta ja erilaisten mittaustietojen kerääminen. Hallintatasolla määritellään säännöt, joiden mukaan datatason SDN-kytkimet reitittävät paketteja. Säännöt välitetään SBI:llä toimivan protokollan avulla SDN-kytkimelle. SDN-ohjaimelta tulevat käskyt voivat lisätä, poistaa tai muokata SDN-kytkimessä olevan vuotaulun merkintöjä. Jos verkossa tapahtuu muutoksia, SDN-kytkin ilmoittaa niistä SDN-ohjaimelle SBI:n kautta ja SDN-ohjain tekee tarvittavat päivitykset SDN-kytkimiin. Verkkotopologian löytämiseen SDN:ssä voidaan käyttää LLDP-protokollaan (Link Layer Discovery Protocol) perustuvaa topologian löytämistä. SDN-ohjain lähettää SDN-kytkimen jokaiselle portille LLDP-paketin, jonka portit lähettävät eteenpäin. Kun SDN-kytkin vastaanottaa LLDP-paketin joltain muulta kuin SDN-ohjaimelta, SDN-kytkin lähettää paketin SDN-ohjaimelle. Kahden solmupisteen välissä siirtyvään LLDP-pakettiin on tallennettu lähettävän ja vastaanottavan SDN-kytkimen tietoja, kuten kytkimet ja portit identifioivat tunnukset. Näiden tietojen perusteella SDN-ohjain pystyy päättelemään, että kahden SDN-kytkimen välissä on yhteys. SDN:n ja perinteisen tietoverkon ero topologian löytämisessä LLDP-protokollalla on se, että SDN:ssä LLDP-paketit prosessoidaan hallintatason SDN-ohjaimessa ja perinteisessä tietoverkossa prosessointi tehdään paketin vastaanottavassa kytkimessä.

Verkkoa hallitsee SDN-ohjain, joka on hallintatasolle sijoittuva ohjelmisto. SDN-ohjaimella on näkymä koko verkosta ja se pystyy hallinnoimaan jokaista verkon SDN-kytkintä. SDN-ohjain voi toimia joko yksin tai hajautettuna [17]. Hajautetussa toteutuksessa verkon hallinta on loogisesti keskitetty, mutta verkkoa hallitsevat SDN-ohjaimet sijaitsevat fyysisesti erillään toisistaan. Verkon hallinnan hajauttaminen lisää turvallisuutta, sillä yhden SDN-ohjaimen varassa oleva verkko voi pudota kokonaan SDN-ohjaimen vioittuessa. Hajautetussa toteutuksessa verkko on myös paremmin

skaalattavissa, koska useampi SDN-ohjain kykenee palvelemaan suurempaa määrää SDN-kytkimiä ja SDN-ohjaimien on mahdollista jakaa kuormaa keskenään. SDN-ohjaimilla on sekä kaupallisia valmistajia (Cisco, HP, Juniper) että avoimen lähdekoodin toteutuksia, kuten OpenDaylight, Floodlight ja Open Network Operating System (ONOS).

Hallintatasolla on yhteys kolmeen eri suuntaan. SBI on hallintatason ja datatason välinen rajapinta, jonka kautta hallintataso välittää käskyjä datatasolle. Datataso käyttää SBI:tä kyselyihin, joilla se hankkii hallintatasolta ohjeita pakettien välittämiseen. Hallintataso välittää ohjeet samaa rajapintaa käyttäen datatasolle. SBI:n kautta kulkee myös hallintatason tarvitsema informaatio verkosta ja datatason laitteiden konfigurointidata. Hallintatason tehtäviä voidaan jakaa niiden aikakriittisyyden perusteella [18]. Nopeaa toimintaa vaatii esimerkiksi vuotaulujen taaja päivittäminen. Hallintataso on yhteydessä sovellustasoon NBI:n välityksellä. Hallintatason east-west-yhteys tarkoittaa hallintatason laitteiden eli SDN-ohjaimien keskinäistä kommunikointia. East-west-yhteyden avulla SDN-ohjaimet voivat jakaa tietojaan verkosta, mikä on tarpeellista esimerkiksi verkkotopologian löytämisessä. East-west-yhteys on looginen yhteys ja todellinen datapakettien siirtäminen tapahtuu datatason kautta.

2.3.3 Sovellustaso

Sovellustasolla sijaitsevat sovellukset, jotka välittävät NBI:n kautta hallintatasolle käskyjä. Käskyt muokkaavat verkkoa halutunlaiseksi. Sovellukset kertovat, mitä ominaisuuksia tai toimintoja verkolta halutaan, ja hallintatason SDN-ohjain muokkaa näiden pyyntöjen mukaan datatason SDN-kytkimiä. Verkon haltija voi sovelluksien kautta määrittää palvelun laadun ja muut verkon ominaisuudet. NBI tarjoaa ohjelmointirajapinnan (Application Programming Interface, API) sovelluksille. Sovellustason sovelluksien ohjelmointiin voidaan käyttää korkean tason ohjelmointikieliä. SDN-arkkitehtuurin yksi vahvuus on abstrakti näkymä, jonka SDN-sovellukset saavat verkosta. Sen avulla verkko voi analysoida itseään ja yhdistää reaaliaikaista tietoa sovelluksiin. Abstrakti näkymä verkosta mahdollistaa verkon resurssien joustavamman käytön.

SDN:ssä sovellustasolle sijoittuvat verkon toiminnot, kuten kuormantasaus, palomuurit ja reititys [19]. Palomuuri on mahdollista toteuttaa asettamalla vuotauluun merkintä, joka estää tietystä osoitteesta tulevan datan päästämisen suojattavaan kohteeseen. Kuormantasaamiselle on suuri tarve teollisuudesta, minkä takia kuormantasaus on myös yksi tärkeimpiä tutkimuskohteita SDN:ssä [20]. Suurissa konesaliverkoissa tarvitaan kuormantasausta minimoimaan verkon ruuhkautuminen. Perinteisissä tietoverkoissa

kuormantasaus on hankalaa, koska verkosta ei ole kokonaista näkymää. Perinteisissä verkoissa käytetyt kuormantasaajat ovat lisäksi kalliita [21]. SDN:n keskitetty hallinta mahdollistaa SDN-ohjaimelle näkymän koko verkosta, ja sovellustasolla tämä näkymä nähdään abstrahoituna. Näin ollen sovellustasolla voidaan tehdä tehokasta kuormatasausta.

2.4 SDN:n käyttö

Internetin valtavan kasvun myötä myös tietoverkkojen ylläpitämiseen tarvittavat järjestelmät kasvoivat, mikä johti tietoverkkojen hallinnan kehittämiseen. Tietoverkkojen ohjelmoitavuutta ja automaattista hallintaa alettiin tutkia ja kehittää 1990-luvun lopun ja 2000-luvun alun aikaan [22]. Nykyään tietoverkkojen koko on edelleen kasvanut verkkoon kytkettyjen mobiililaitteiden ja pilvipalveluiden myötä. Suuret datakeskukset ja organisaatioiden verkkojen väliset yhteydet vaativat entistä tehokkaampaa hallintaa ja automaatiota. Vuonna 2008 julkaistu tutkimus [23] esitteli uuden tietoverkkojen testaamiseen tarkoitetun OpenFlow-protokollan. OpenFlow'n myötä kehittyi ajatus SDN:stä. SDN:ää voidaan hyödyntää esimerkiksi datakeskuksissa, akateemisessa tutkimuksessa ja mobiiliverkoissa.

Yksi potentiaalisimpia käyttökohteita SDN:lle on datakeskukset. Datakeskuksissa voi olla satoja palvelimia, joita yhdistetään sadoilla kytkimillä. Tällaisen verkkolaitemäärän hallintaan tarvitaan automaatiota ja keskitettyä hallintaa. SDN:n yhteydessä puhutaan usein tietoverkkojen virtualisoinnista eli NFV:stä (Network Functions Virtualization). NFV tarkoittaa tietoverkon solmupisteen toimintojen, kuten reitittämisen, kuorman tasauksen tai palomuuritoiminnan erottamista sen fyysisestä toteutuksesta. Kokonaisten tietoverkkojen virtualisointi mahdollistaa usean eri virtuaaliverkon olemisen samassa fyysisessä verkossa. Verkkojen virtualisointia voidaan hyödyntää datakeskuksissa. SDN ja NFV eivät ole toisistaan riippuvaisia mutta niiden yhtäaikainen käyttö parantaa verkon muokattavuutta. Kuorman tasaaminen on tärkeä osa datakeskuksia, ja sitä voidaan tehostaa käyttämällä SDN:ää [24]. Esimerkki SDN:ää hyödyntävästä järjestelmästä on Googlen datakeskuksia yhdistävä B4-tietoverkkojärjestelmä [25].

SDN muodostui alun perin tietoverkkojen testaamiseen ja tutkimukseen tarkoitetun OpenFlow-protokollan seurauksena. OpenFlow soveltui tutkimuskäyttöön, koska se mahdollisti uusien protokollien testaamisen olemassa olevissa verkoissa. Käytössä olevassa verkossa testaaminen poistaa tarpeen erillisen testiverkon muodostamisesta. Lisäksi voidaan varmistua uusien protokollien toimivuudesta myös oikeissa verkoissa sen sijaan, että niiden toimintaa testattaisiin vain erillisissä testiverkoissa.

Tutkimuksessa [26] on selvitetty SDN:n tuomia hyötyjä mobiiliverkoissa. Tutkimuksessa todetaan, että solujen välistä häiriötä voidaan pienentää SDN:llä. Keskitetyn hallinnan ansiosta verkolla on tieto sen globaalista tilasta, jolloin radioresursseja voidaan jakaa optimoidusti. Tutkimuksessa mainitaan SDN:n tuomiksi hyödyiksi myös mobiilitietoliikenteen hallinta ja verkon virtualisointi. Esimerkiksi palvelun laadun periaatteita on huomattavasti parempi hallita SDN:ssä. Verkon virtualisoinnilla taas voidaan yksinkertaistaa ja tehostaa mobiiliverkon laitteistoa.

3. SDN-PROTOKOLLAT

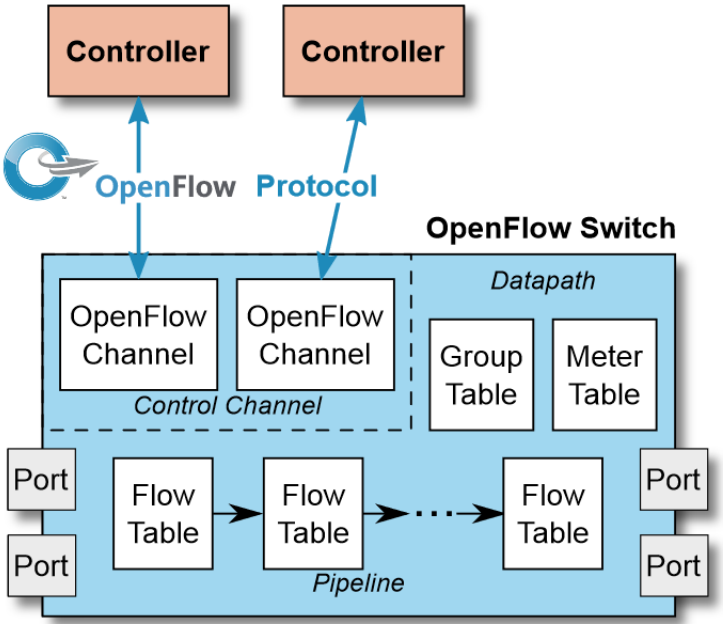
Protokollia käytetään tietoliikenteessä yhteyksien muodostamiseen ja yhteisten sääntöjen määrittämiseen. SDN:ssä protokollia tarvitaan datatason ja hallintatason yhdistämiseen sekä datatason laitteiden konfigurointiin. Tässä luvussa käydään läpi SDN:ssä käytettäviä protokollia. Tässä työssä tarkasteltavat protokollat on jaettu kahteen ryhmään. Mendiola et al. [27] käyttävät termejä D-CPI-protokolla (Data-Controller Plane Interface) ja MI-protokolla (Management Interface), joita käytetään myös tässä työssä. D-CPI-protokollat toimivat hallintatason ja datatason välisenä yhteytenä. Datatason laitteita konfiguroidaan MI-protokollalla. Sekä D-CPI- että MI-protokollat voidaan ajatella toimivan SBI-rajapinnassa.

3.1 D-CPI-protokollat

SDN:n perusajatus on erottaa hallintataso ja datataso toisistaan. Näiden tasojen välistä yhteyttä varten tarvitaan protokolla. D-CPI-protokollat välittävät viestejä hallintatason ja datatason välillä. Hallintatasolta tulevat viestit muokkaavat datatasolla olevia pakettien välityskäytäntöjä. Toisaalta voidaan ajatella siis, että D-CPI-protokollat muodostavat SDN:n perustan. D-CPI-protokolliin liittyy yleensä myös malli (engl. *framework*). Tässä alaluvussa käsiteltävät protokollat ovat siis oikeastaan kokonaisuuksia, jotka sisältävät sekä protokollan että mallin datatason rakentumisesta ja datatason eri rajapinnoista.

3.1.1 OpenFlow

SDN kehitettiin alun perin kampusverkon testaamiseen ja uusien protokollien kehittämiseen [23]. Verkon hallinnan mahdollisti OpenFlow-protokolla [28], joka on perustana koko SDN:n ajatukselle. OpenFlow on nykyisin käytetyin SDN-protokolla ja usein SDN:ää ja OpenFlow'ta pidetään samana asiana. Todellisuudessa SDN tarkoittaa ohjelmoitavaa verkkorakennetta, jossa hallintataso ja datataso on erotettu toisistaan. OpenFlow on vain yksi tapa toteuttaa SDN:ää. [13] OpenFlow on protokolla, joka välittää hallintatasolta viestejä datatasolle ja määrittää pakettien välittämisen datatasolla. Hallintatasolta tulevat viestit ovat vuotaulujen merkintöjä sekä kyselyitä kytkimen tilasta. OpenFlow'ta tukevia kytkimiä kutsutaan OpenFlow-kytkimiksi. OpenFlow-kytkimen sisältö on kuvattu kuvassa 5. Luvussa 2.2 esitelty SDN:n toiminta vuotauluihin perustuen on hyvin samankaltainen OpenFlow'n toiminnan kanssa



Kuva 4. OpenFlow-kytkin. [28]

Kuvan 4 mukaisesti OpenFlow-kytkin sisältää fyysiset portit (Port) saapuville ja lähteville paketeille, yhden tai useamman vuotaulun (Flow Table), yhden tai useamman kanavan (OpenFlow Channel) hallintatason ohjaimiin, ryhmätaulun (Group Table) sekä mittataulun (Meter Table).

Vuotaulut koostuvat vuomerkinnöistä (Flow Entry), jotka ovat joukko kenttiä, joihin saapuvaa pakettia verrataan.

Match Fields	Priority	Counters	Instructions	Timeouts	Cookie	Flags

Kuva 5. Vuomerkintä. [28]

Kuvassa 5 on vuotaulun yhden vuomerkinnän sisältämät kentät. Vuomerkinnän Match Fields -kenttää verrataan saapuneen paketin otsikkoon. Vuotauluilla ja vuomerkinnöillä on prioriteetti (Priority), jonka mukaan muodostuu järjestys, jossa paketin tietoja vertaillaan. Alhaisimmalla prioriteetilla on puuttuva vastaavuus -merkintä (engl. *table-miss*), joka määrittää, mitä paketille tehdään, kun vuotaulussa ei ole yhtäkään osumaa kyseiselle paketille. Paketti voidaan esimerkiksi pudottaa, lähettää seuraavalle vuotaululle tai lähettää hallintatason ohjaimelle. Vuotauluja voi olla OpenFlow-kytkimellä useampia, jolloin pakettia siirretään järjestyksessä vuotaulusta toiseen, kunnes vastaavuus löytyy. Counters-kenttä sisältää OpenFlow-kytkimen laskureita, kuten vastaanotettujen pakettien ja tavujen määrä. Jos paketille löytyy vastaavuus Match Field

-kentässä, sitä käsitellään Instructions-kentän ohjeiden mukaisesti. Paketti voidaan esimerkiksi pudottaa, sen otsikkoa voidaan muokata tai se voidaan ohjata vuotauluputkiston seuraavaan vuotauluun. Timeouts-kentässä ilmoitetaan maksimiaika, jonka kytkin pitää vuota elossa. Cookie-kenttä on SDN-ohjaimen käyttämä kenttä, jota voidaan käyttää vuon suodattamiseen vuon tilastojen ja muutosten perusteella. Tätä kenttää ei käytetä tavallisessa paketin prosessoinnissa. Flags-kenttää käytetään vuomerkintöjen hallinnassa.

Ryhmätaulu on vuotauluputkiston päässä oleva taulu, jossa putkiston läpi tulleen paketin lähettämistä voidaan käsitellä. Paketti voidaan esimerkiksi kopioida ja kopiot voidaan merkitä eri VLAN-tunnuksilla, jolloin ne voidaan lähettää eri verkkoihin tai paketti voidaan yleislähettää (engl. *broadcast*), jolloin paketin kopioita lähetetään jokaiseen porttiin. Mittataulun merkinnät määrittävät vuokohtaisesti pakettien siirtonopeuteen liittyvät ominaisuudet. Jokainen vuomerkintä viittaa johonkin mittataulun merkintään ja paketin siirto-ominaisuudet määritetään sen mukaan.

OpenFlow-kanavan kautta SDN-ohjain päivittää OpenFlow-kytkimen vuotauluja, ja SDN-ohjain sekä OpenFlow-kytkin välittävät paketteja toisilleen. Ohjaimen ja kytkimen väliset viestit voivat olla ohjaimen tai kytkimen aloitteesta lähetettäviä. Ohjaimelta kytkimelle lähtevät viestit (Controller-to-Switch) voivat olla kyselyitä kytkimen ominaisuuksista ja asetuksista, useamman ohjaimen hallintaan liittyviä viestejä tai vuotaulujen päivitysviestejä. Epäsynkronisia (Asynchronous) viestejä ovat kytkimen aloitteesta lähtevät viestit. Kytkimen lähettämiä viestejä ovat esimerkiksi table-miss -tapahtumasta aiheutuvat kyselyt, kytkimen porteissa tapahtuvista muutoksista tiedottaminen ja uusiin ohjaimiin kytkeytymisestä tiedottaminen. Symmetriset viestit ovat joko ohjaimen tai kytkimen aloitteesta lähteviä viestejä. Tällaisia ovat esimerkiksi Hello- ja Echo-viestit. OpenFlow-kanavassa kulkevat viestit kulkevat TCP- ja TLS-protokollien (Transmission Control Protocol, Transport Layer Security) päällä.

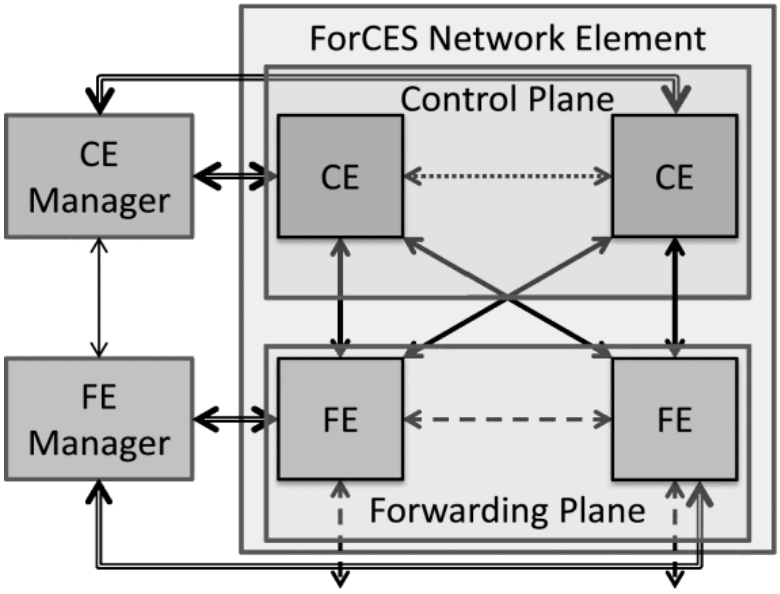
OpenFlow’n avulla tietoverkkoa voidaan hallita hyvin yksityiskohtaisesti. Vuotaulujen vuomerkintöjen hienojakoinen säätäminen mahdollistaa datatason abstraktin näkymän hallintatasolle, jolloin datatasolla voidaan toteuttaa monimutkaisiakin toimintoja. OpenFlow’ssa SDN-ohjaimella on laaja näkemys koko verkosta, minkä takia verkon muutoksiin, kuten linkkien katkeamiseen voidaan reagoida nopeasti ja helposti [13]. Samoin esimerkiksi uusien palvelun laadun periaatteiden asettaminen verkkoon on OpenFlow’lla yksinkertaista. OpenFlow on SDN-protokollista eniten käytössä, ja monet valmistajien sekä avointen lähdekoodien toteutukset tukevat sitä. OpenFlow on avoin rajapinta datatason ja hallintatason välillä, minkä takia palveluntarjoajat voivat kehittää

omia OpenFlow-yhteensopivia SDN-ohjaimia. OpenFlow'n yhtenä ongelmana voidaan pitää data- ja hallintatason välisen viestinnän aiheuttamaa yleisrasitetta. Data- ja hallintatason välinen yhteys vaikeuttaa siten myös skaalautuvuutta, koska SDN-ohjaimeen yhteydessä olevien OpenFlow-kytkimien määrän kasvu lisää yleisrasitetta. OpenFlow'ssa käytetty verkon keskitetty hallinta on tietoturvan kannalta ongelma. Yhdellä SDN-ohjaimella toteutettu verkko kaatuu, jos SDN-ohjain kaatuu. Koska keskitetyllä SDN-ohjaimella on näkymä koko verkosta, hyökkääjän saadessa ohjaimen haltuunsa koko verkko on haavoittuvainen.

3.1.2 ForCES

Vuonna 2003 IETF:n (Internet Engineering Task Force) alaisuudessa oleva ForCES-työryhmä kehitti ForCES-mallin (Forwarding and Control Element Separation). ForCES-malli ei kuitenkaan saanut hyväksyntää alalla ja se hylättiin vuosiksi. SDN:n nousun myötä ForCES-työryhmä aloitti mallin standardoinnin uudelleen. ForCES sisältää mallin siitä, miten dataa siirretään sekä protokollan, jolla mallin osapuolet kommunikoivat keskenään. ForCES:n datansiirtomalli on kuvattu dokumentissa [29] ja ForCES-protokolla dokumentissa [30]. ForCES on verkkomalli, jossa välittävät elementit (engl. *Forwarding Element*, FE) ja hallintaelementit (engl. *Controlling Element*, CE) on erotettu toisistaan.

ForCES-mallissa verkkoelementit (engl. *Network Element*, NE) vastaavat perinteisten verkkojen reitittimiä. NE sisältää yhden tai useamman CE:n sekä yhden tai useamman FE:n. CE:t hallitsevat FE:itä määrittämällä ForCES-protokollalla niille ohjeet, miten paketteja tulee välittää. ForCES on siis isäntä-orja-tyyppinen malli, jossa CE:t ovat isäntiä ja FE:t orjia. CE:t ajatellaan kuuluvan SDN:n hallintatasolle ja FE:t datatasolle. ForCES-protokolla on CE:iden ja FE:iden välistä informaatiota välittävä protokolla. ForCES-malli kokonaisuudessaan voi sisältää useampia protokollia. Esimerkiksi CE:iden välinen kommunikointi ei kuulu ForCES-protokollan tehtäviin, vaan sitä varten tarvitaan muita protokollia.



Kuva 6. ForCES-malli. Muokattu lähteestä [31]

Kuvassa 6 on ForCES-mallin looginen toteutus. Kuvasta nähdään, että NE:n sisällä CE:t on sijoitettu hallintatasolle ja FE:t datatasolle. ForCES:n yhteydessä käytetään datatasosta termiä välitystaso (engl. *forwarding plane*). Kuvassa 6 on kuvattuna myös hallintaelementtien haltija (CE Manager) sekä välittävien elementtien haltija (FE Manager). Nämä haltijat alustavat FE:t ja CE:t ennen kuin ne ovat yhteydessä toisiinsa. Alustusvaiheessa (Pre-association Phase) elementtien haltijat määrittelevät, mitkä FE:t ja CE:t kuuluvat samaan NE:hen. Tässä vaiheessa määritellään myös tarvittavat attribuutit, kuten IP-osoitteet, FE:iden ja CE:iden välisen yhteyden muodostamiseen. Elementtien haltijat voivat sijaita sulautettuna niitä vastaavissa elementeissä tai fyysisesti ja loogisesti erillään. CE:iden ja FE:iden välisten yhteyksien määrä on valmistajien päätettävissä. Yhteen FE:hen voi olla yhteydessä useampi CE samoin kuin yhteen CE:hen voi olla yhteydessä useampi FE. Useamman kuin yhden CE:n käyttö parantaa verkkoelementin luotettavuutta ja mahdollistaa esimerkiksi kuorman tasaamisen.

FE koostuu LFB:istä (Logical Functional Block). LFB on abstrakti kuvaus jostakin paketin käsittelyyn liittyvästä resurssista. LFB:t ovat tarkoin rajatuista toiminnoista koostuvia rakennuspalikoita, jotka yhdessä muodostavat FE:n. LFB:t muodostavat topologian FE:n sisällä ja niitä yhdistelemällä on mahdollista suorittaa monimutkaisiakin paketin välitykseen liittyviä toimintoja. LFB:iden ominaisuudet näkyvät hallintatason CE:lle, joka voi konfiguroida niitä. [31]

ForCES eroaa OpenFlow'sta eniten siinä, että sen hallinta ei ole keskitetty loogisesti yhteen paikkaan. ForCES:ssa uusien datatason toimintojen käyttöönotto on helpompaa kuin OpenFlow'ssa, jossa OpenFlow-kytkimen toiminnot rajoittavat itse protokollan toimintaa. ForCES-mallissa verkkolaitteet voidaan konfiguroida suoraan ilman erillistä MI-protokollaa toisin kuin OpenFlow'ssa, jossa OpenFlow-kytkin täytyy konfiguroida siihen tarkoitukseen olevalla protokollalla. OpenFlow on kuitenkin teollisuudessa laajemmin käytetty kuin ForCES. ForCES:sta ei myöskään ole avoimia toteutuksia helposti saatavilla, mikä vaikeuttaa sen tutkimusta ja testaamista. [31]

3.1.3 BGP-LS/PCEP

SDN voidaan toteuttaa yhdistämällä kaksi perinteisissä tietoverkoissa käytettyä protokollaa. BGP-LS (Border Gateway Protocol Link State) [31] ja PCEP (Path Computation Element Communication Protocol) [32] -protokollat yhdessä muodostavat SDN:n tarvitseman pohjan [26, 33-35]. Tässä toteutuksessa PCE-malli (Path Computation Element) ja PCEP-protokolla vastaavat verkon reitityksen laskennasta. PCE ei yksinään pysty keräämään tietoa verkon tilasta, joten tuekseen se tarvitsee BGP-LS-protokollan, joka välittää sille tietoa verkon sisäisten ja ulkoisten linkkien tiloista.

PCE-malli koostuu PCE-verkkoelementistä ja PCC-asiakkaasta (Path Computation Client). PCE on verkon laskentaa suorittava elementti. PCE:itä voi olla samassa verkossa myös useampi kuin yksi. Suoritettava laskenta voidaan jakaa eri PCE:iden välillä tai yksi PCE voidaan asettaa vastaamaan laskennasta ja toinen PCE varmistamaan verkon säilyminen, mikäli ensisijainen PCE kaatuu. PCE:n suorittama laskenta tapahtuu PCC:n pyynnöstä. PCC on verkossa oleva paketteja välittävä elementti, kuten reititin. PCE käyttää laskennassaan TED-tietokantaa (Traffic Engineering Database). TED sisältää tiedot verkkojen linkeistä ja niiden tiloista, ja PCE käyttää näitä tietoja laskennassaan. TED sisältää kuitenkin vain paikallisen verkon linkkien tiedot. PCEP on TCP-protokollan päällä toimiva protokolla, joka vastaa PCE:n ja PCC:n välisestä kommunikoinnista. Useamman PCE:n järjestelmässä PCEP vastaa myös PCE:iden välisestä kommunikoinnista. PCE mudostaa PCC:iden välille LSP:itä (Label Switched Path). LSP on polku, jonka kautta kulkee samoin tunnuksin varustetut paketit. LSP voidaan ajatella BGP-LS/PCEP-toteutuksen vastineena vuolle.

BGP on autonomisten järjestelmien välillä toimiva protokolla, joten sen avulla PCE:lle voidaan välittää tietoja myös PCE:n ulkopuolisista verkoista. BGP välittää reititys- ja saavutettavuustietoja verkkojen välillä ja yksittäisen verkon sisällä. BGP-LS on BGP:n laajennus, jonka avulla voidaan jakaa eri tietoja tietoverkon linkkien tiloista. Reitittimet

ylläpitävät tietokantoja niiden läheisyydessä olevista linkeistä. Tietokannoissa säilytetään tietoja muun muassa IP-osoitteista, linkkien kaistanleveyksistä ja varattavissa olevista linkkien kaistanleveyksistä. BGP-LS:n avulla tietokannoissa olevia tietoja linkkien tiloista ja verkosta kerättyjä topologiatietoja voidaan jakaa eteenpäin.

BGP-LS/PCEP-toteutuksella ei ole mahdollista tehdä yhtä hienojakoista verkon ohjelmointia kuin esimerkiksi OpenFlow'lla ja ForCES:lla. Skaalautuvuuden näkökulmasta BGP-LS-protokolla ei tarvitse uusia toimijoita, kun verkon kokoa kasvatetaan. Toisaalta suurissa verkoissa BGP-LS:n välittämän datan määrä voi kasvaa suureksi. [27] BGP-LS/PCEP-toteutuksesta parhaan hyödyn verrattuna muihin SDN-toteutuksiin saa, kun se toteutetaan perinteisen tietoverkon päälle. BGP-LS/PCEP voi käyttää verkkolaitteinaan samoja kytkimiä ja reitittimiä kuin perinteiset verkot. Erona ForCES- ja OpenFlow-protokolliin PCEP pystyy välittämään viestejä myös hallintatason elementtien välillä.

3.2 MI-protokollat

D-CPI-protokollat määrittävät, miten datapaketit reititetään tietoverkossa. Ne eivät kuitenkaan ota kantaa verkkolaitteiden asetuksiin. Verkkolaitteiden ominaisuuksien – kuten IP-osoitteiden, porttien sekä pakettijonojen (engl. *queue*) – konfigurointia varten tarvitaan omia protokollia. Yksi puhutuimpia tietoverkkoihin liittyviä ongelmia on tietoverkkojen hallinnan automatisointi. Tämä tarkoittaa, että verkkolaitteiden konfigurointia pyritään automatisoimaan. MI-protokollat vastaavat verkkolaitteiden konfiguroinnista.

3.2.1 OF-Config

OpenFlow-protokolla määrittää OpenFlow-kytkimen vuotaulut mutta se ei määritä varsinaisesti itse kytkimen konfigurointia. OF-Config (OpenFlow Management and Configuration protocol) [37] on OpenFlow-kytkimen konfigurointia varten kehitetty protokolla, joka määrittää OpenFlow-kytkimen resurssit, kuten portit, IP-osoitteet ja pakettijonot. OF-Config on OpenFlow'ta tukeva protokolla, ja se käyttää kuljetusprotokollanaan NETCONF-protokollaa. OF-Config toimii etäkäyttöisenä ja sen operointipiste (OpenFlow Configuration Point) sijaitsee erillään kytkimestä. Standardi ei ota kantaa siihen, missä operointipiste sijaitsee. Se voi olla esimerkiksi hallintatason ohjaimeen yhdistetty ohjelmisto tai verkonvalvojalla.

OF-Config:lla on standardissa määriteltyjä vaatimuksia liittyen OpenFlow-kytkimen konfigurointiin. Näitä vaatimuksia ovat esimerkiksi kytkimen aloitteesta muodostettavaan

yhteyteen vaadittavien parametrien konfigurointi, katkenneesta yhteydestä aiheutuvan poikkeustilan konfigurointi ja sertifikaatin konfigurointi kytkimelle yhteyden salausta varten. OF-Config:n tarkoitus on antaa OpenFlow-kytkimelle parametrit, joita se tarvitsee näiden vaatimusten täyttämiseen. SDN-kytkimen ja SDN-ohjaimen välisen yhteyden muodostamiseen tarvittavia parametreja ovat SDN-ohjaimen IP-osoite, SDN-kytkimen ja SDN-ohjaimen väliseen yhteyteen käytetty kuljetusprotokolla, ohjaimen päässä käytettävä portti sekä kytkimen päässä käytettävä portti, mikäli yhteys muodostetaan ohjaimen aloitteesta.

OF-Config:lla on mahdollista muodostaa ja hallita pakettijonoja, mikä parantaa palvelun laadun hallintaa. Alun perin OpenFlow'ta varten kehitetty OF-Config osaa hyödyntää OpenFlow'n ominaisuuksia tehokkaasti. Jos OF-Config:n operointipiste ja SDN-ohjain sijaitsevat erillään, niiden välillä oleva kommunikointi täytyy toteuttaa erikseen, jotta ne voidaan synkronoida. [27] OF-Config:n avulla ei saada kerättyä kaikkia tietoja laitteiston resursseista, kuten suorittimen ja muistin käytöstä [38].

3.2.2 OVSDB-hallintaprotokolla

Virtuaalikoneiden väliseen kommunikointiin sekä virtuaalikoneiden yhdistämiseen fyysiseen verkkoon käytetään virtuaalikytkimiä. Yksi käytetyimpiä virtuaalikytkimiä on avoimen lähdekoodin Open vSwitch (OVS) [39]. OVS on alun perin Niciran kehittämä Apache 2.0 lisenssin alaisuudessa toimiva, ohjelmistona toteutettu kytkin. OVS tukee OpenFlow'ta ja se on laajasti käytössä SDN-toteutetuissa konesaliverkoissa. OVS ei riipu hallintatason SDN-ohjaimesta ja sitä voidaan käyttää myös ilman SDN-ohjainta perinteisen kytkimen tapaan.

OVS sisältää tietokannan, jossa säilytetään kytkimen konfigurointiin liittyvää dataa. OVS:n resurssien hallintaan on kehitetty OVSDB-hallintaprotokolla (Open vSwitch Database Management Protocol). OVSDB-hallintaprotokollan avulla voidaan hallita virtuaalikytkimen resursseja etäkäyttönä. Sen avulla voidaan esimerkiksi konfiguroida portteja, tunneleita sekä hallita QoS-periaatteita. OVSDB:n hallintayksikkö on OVSDB-hallintaprotokollan välityksellä yhteydessä OVSDB-palvelimeen, joka sisältää JSON-formaatissa (JavaScript Object Notation) esitetyn, kytkimen tilaa kuvaavan tietokannan. [40]

OF-Config:n tapaan OVSDB-hallintaprotokollalla voidaan luoda ja muokata pakettijonoja, mikä tekee siitä tehokkaan protokollan palvelun laadun hallintaan. Volpato et al. [41] esittelevät tutkimuksessaan palvelun laadun hallintaa autonomisesti

suorittavan ohjelman, joka käyttää datatasonhallintaan OVSDB-hallintaprotokollaa. Tutkimuksessaan he toteavat, että tällä ohjelmalla voidaan vähentää pakettihäviöitä, optimoida resursseja ja parantaa palvelun laatua merkittävästi. OVSDB-hallintaprotokollan heikkous on se, että sitä voidaan käyttää ainoastaan OVS-ympäristössä.

3.2.3 NETCONF

Vuonna 2002 Internet Architecture Board:n (IAB) järjestämässä työpajassa tietoverkko-operaattorit ja tietoliikenneprotokollien kehittäjät kokoontuivat pohtimaan tietoverkkojen hallinnan kehittämistä. Yksi työpajassa muodostettu ehdotus oli IETF:n resurssien käyttäminen tietoverkkojen hallintamekanismin standardointiin. [42] Seuraavana vuonna IETF perusti työryhmän, jonka tarkoitus oli kehittää tietoverkkojen hallintaan tarkoitettu protokolla. Työryhmän toiminta johti Network Configuration Protocol -protokollan (NETCONF) julkaisuun vuonna 2006 [43]. NETCONF on protokolla, joka tarjoaa API:n, jolla sovellukset voivat hallita verkkolaitetta sekä hakea ja päivittää sen konfigurointidataa.

NETCONF voidaan jakaa neljään kerrokseen: Content, Operations, Message ja Secure Transport. Content-kerros sisältää protokollan käsittelemän datan XML-muodossa (Extensible Markup Language). XML on helposti ymmärrettävää ja sitä voi tulkita sekä ihmiset että koneet. XML on laajasti käytössä esimerkiksi web-palveluissa. Operations-kerroksella määritellään protokollan perustoiminnot. Perustoimintoihin kuuluvat esimerkiksi <get>-toiminto, joka hakee käytössä olevan konfigurointidatan ja tiedot laitteen tilasta sekä <edit-config>-toiminto, jolla voidaan luoda, poistaa tai korvata konfigurointitietoa. Message-kerros tarjoaa RPC-mekanismin (Remote Procedure Call), jolla Operations-kerroksen toiminnot sovelluksen ja laitteen välillä voidaan toteuttaa. RPC:n avulla eri nimiavaruuksissa toimivat sovellus ja verkkolaite voivat kommunikoida toisilleen. Secure Transport -kerros on NETCONF:n kuljetuskerros. Standardin mukaan kuljetusprotokollan on oltava yhteydellinen ja sen on tarjottava siirrettävän datan autentikointi, eheys ja luottamuksellisuus. Lisäksi NETCONF-toteutuksen on tuettava SSH-protokollaa. NETCONF voi siis toimia minkä tahansa suojatun, yhteydellisen kuljetuskerroksen protokollan yli. Yleisimmin käytetään SSH-yhteyttä. [43]

NETCONF on valmistajasta riippumaton ja monet verkkolaitevalmistajat ovat ottaneet sen käyttöön laitteissaan. Valmistajienvälisen yhteensopivuuden mahdollistaa XML-merkintäkielen ja RPC-mallin käyttäminen. Varsinkin SDN-mallissa, jossa verkon SDN-kytkimet ja SDN-ohjaimet voivat olla eri valmistajilta, konfigurointiprotokollan

yhteensopivuus eri valmistajien laitteiden välillä on tärkeä ominaisuus. NETCONF:n standardissa vaaditaan kuljetuskerroksen suojattua yhteyttä, mikä tarkoittaa, että NETCONF:a käyttävät sovellukset voivat olla varmoja tiedonsiirron turvallisuudesta.

4. YHTEENVETO

Tietoverkkojen kasvaminen ja verkkorakenteiden kompleksisuus ovat aiheuttaneet tarpeen kehittää tietoverkkojen hallintaa ja automatisointia. Ohjelmisto-ohjattu tietoverkko eli SDN on ajatusmalli, jossa datapakettien välityksestä vastaava datataso ja verkon hallintaan liittyvästä toiminnasta vastaava hallintataso on erotettu toisistaan. Tämän erotuksen seurauksena verkosta saadaan abstrakti, koko verkon laajuinen näkymä, jota voidaan hallita joustavasti. SDN soveltuu varsinkin suuria verkkolaitemääriä sisältäviin tietoverkkoihin. Data- ja hallintatason lisäksi SDN:ään kuuluu sovellustaso, jossa sijaitsevat verkon toimintoja ohjaavat SDN-sovellukset. Hallintataso näyttäytyy myös abstraktina näkymänä sovellustasolle, minkä ansiosta voidaan tehdä monimutkaisiakin toimintoja verkkoon.

SDN tarvitsee toimiakseen yhteyskäytännön hallintatason ja datatason välille. SDN:ssä käytettävät protokollat voidaan luokitella niiden tehtävien mukaan D-CPI- ja MI-protokolliin. D-CPI-protokollat viestivät hallintatason SDN-ohjaimen ja datatason SDN-kytkinten välillä sekä määrittävät SDN-kytkimille ohjeet pakettien reitittämiseksi. MI-protokollia käytetään SDN-kytkinten omien ominaisuuksien konfigurointiin. Yhteinen piirre eri D-CPI-protokollilla on erilliset elementit verkon hallintaan liittyvälle laskennalle ja pakettien välittämiselle datatasossa. Eroavaisuuksia D-CPI-protokollissa on melko paljon, koska ne toimivat erilaisissa malleissa. Eri D-CPI-protokollien paremmuudesta ei voida tehdä päätelmiä, koska protokollista ainoastaan OpenFlow on saanut paljon huomiota teollisuudessa ja tutkimuksessa. ForCES- ja BGP-LS/PCEP-toteutukset ovat olleet verrattain vähän tutkimuksen kohteena. MI-protokollissa ei ole suurta eroa. Kaikkien MI-protokollien toimintona on asettaa SDN-kytkimelle ominaisuudet, joita ei voida asettaa D-CPI-protokollilla. MI-protokollien erona on lähinnä se, mihin ympäristöön ne on alun perin suunniteltu.

OpenFlow’n lisäksi myös muiden protokollien käyttöä tulisi tutkia enemmän. OpenFlow’n suosikkiasema tämän hetken SDN-markkinoilla voi johtaa siihen, että valmistajat eivät enää kiinnostu löytämään uutta tapaa toteuttaa SDN:ää. Tällöin on vaarana, että yleiseen käyttöön tuleva SDN-malli ei ole paras mahdollinen. ForCES-mallin avoimen lähdekoodin toteutuksia tulisi tehdä saataville, jotta sen soveltuvuutta SDN-toteutukseksi voitaisiin arvioida paremmin. MI-protokollien kohdalla on vielä vaikea sanoa, mikä ratkaisu on lopulta paras, koska siihen vaikuttaa se, minkälaisessa SDN-ympäristössä MI-protokollia tullaan lopulta käyttämään.

LÄHTEET

[1] Sherry J, Hasan S, Scott C, Krishnamurthy A, Ratnasamy S, Sekar V, Making middleboxes someone else's problem: network processing as a cloud service, Proceedings of the ACM SIGCOMM 2012 conference on applications, technologies, architectures, and protocols for computer communication, 2012, pp. 13-24

[2] Misra S, Goswami S, Network routing: fundamentals, applications and emerging technologies, Wiley, 2017, pp. 47-53

[3] Hendrick C, Routing Information Protocol, RFC 1058, 1988, https://tools.ietf.org/html/rfc1058

[4] Moy J, OSPF Version 2, RFC 2328, 1998, https://tools.ietf.org/html/rfc2328

[5] Rekhter Y, Li T, Hares S, A Border Gateway Protocol 4 (BGP-4), RFC 4271, 2006, https://tools.ietf.org/html/rfc4271

[6] Kreutz D, Ramos F, Veríssimo P, Rothenberg C, Azodolmolky S, Uhlig S, Software-Defined Networking: A Comprehensive Survey, Proceedings of the IEEE, 2015, pp. 14-76

[7] Karakus M, Durresi A, Quality of Service (QoS) in Software Defined Networking (SDN): A survey, Journal of Network and Computer Applications, 2017, pp. 200-218

[8] Hoang D, Pham M, On software-defined networking and the design of SDN controllers, Proceedings of the 2015 6th International Conference on the Network of the Future (NOF), 2015, pp. 1-3

[9] Kim H, Feamster N, Improving network management with software defined networking, IEEE Communications Magazine, 2013, pp. 114-119

[10] Datacomm: Software Defined Network, 2017, Saatavilla: https://www.datacomm.co.id/en/telco/sdn/, 31.5.2019

[11] Mirchev A, Survey of Concepts for QoS improvements via SDN, Future Internet (FI) and Innovative Internet

Technologies and Mobile Communications (IITM), 2015, pp. 33-40

[12] Lara A, Kolasani A, Ramamurthy B, Network Innovation using OpenFlow: A Survey, IEEE Communications Surveys & Tutorials, 2014, pp. 493-512

[13] Appel J, Dogan C, Fuetsch A, Guanglu S, Howald R, Kashiwa D, Lopez P, McKeown N, Parulkar G, Open Networking Foundation: Strategic plan, 2018, Saatavilla: https://www.opennetworking.org/mission/, 8.8.2019

[14] Banse C, Rangarajan S, A Secure Northbound Interface for SDN Applications, Proceedings of the IEEE, 2015, pp. 834-839

[15] Luo S, Yu H, Li L, Practical flow table aggregation in SDN, Computer Networks, 2015, pp. 72-88

[16] Singh D, Ng B, Lai Y, Lin Y, Seah W, Modelling Software-Defined Networking: Software and hardware switches, Journal of Network and Computer Applications, 2018, pp. 24-36

[17] Oktian Y, Lee S, Lee H, Lam J, Distributed SDN controller system: A survey on design choice, Computer Networks, 2017, pp. 100-111

[18] Haleplidis E, Pentikousis K, Denazis S, Hadi Salim J, Meyer D, Koufopavlou O, Software-Defined Networking (SDN): Layers and Architecture Terminology, RFC 7426, 2015, https://tools.ietf.org/html/rfc7426

[19] Trois C, Del Fabro M, de Bona L, Martinello M, A Survey on SDN Programming Languages: Toward a Taxonomy, IEEE Communications Surveys & Tutorials, 2016, pp. 2687-2712

[20] Zope N, Pawar S, Saquib Z, Firewall and load balancing as an application of SDN, Proceedings of the IEEE, 2016, pp. 354-359

[21] Long H, Shen Y, Guo M, Tang F, LABERIO: Dynamic load-balanced Routing in OpenFlow-enabled Networks, Proceedings of the IEEE, 2013, pp. 290-297

[22] Goransson P, Black Chuck, Software defined networks: a comprehensive approach, Elsevier, 2014, p. 40

[23] McKeown N, Anderson T, Balakrishnan H, Parulkar G, Peterson L, Rexford J, Shenker S, Turner J, OpenFlow: Enabling innovation in campus networks, ACM SIGCOMM COMPUTER COMMUNICATION REVIEW, 2008, pp. 69-74

[24] Tariq E, Ameer H, Mohammed A, Load Balance in Data Center SDN Networks, International Journal of Electrical and Computer Engineering, 2018, p. 3086

[25] Jain S, Kumar A, Mandal S, Ong J, Poutievski L, Singh A, Venkata S, Wanderer J, Zhou J, Zhu M, B4: experience with a globally-deployed software defined wan, Proceedings of the ACM, 2013, pp. 3-14

[26] Tomovic S, Pejanovic-Djurisic M, Radusinovic I, SDN Based Mobile Networks: Concepts and Benefits, Wireless Personal Communications, 2014, pp. 1629-1644

[27] Mendiola A, Astorga J, Jacob E, Higuero M, A Survey on the Contributions of Software-Defined Networking to Traffic Engineering, IEEE Communications Surveys & Tutorials, 2017, pp. 918-953

[28] OpenFlow Switch Specification version 1.5.1, ONF Open Networking Foundation, 2015, https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf

[29] Yang L, Dantu R, Anderson T, Gopal R, Forwarding and Control Element Separation (ForCES) Framework, RFC 3746, 2004, https://tools.ietf.org/html/rfc3746

[30] Doria A, Hadi Salim J, Haas R, Khosravi H, Wang W, Dong L, Gopal R, Halpern J, Forwarding and Control Element Separation (ForCES) Protocol Specification, RFC 5810, 2010, https://tools.ietf.org/html/rfc5810

[31] Haleplidis E, Salim J, Halpern J, Hares S, Pentikousis K, Ogawa K, Weiming W, Denazis S, Koufopavlou O, Network Programmability With ForCES, Proceedings of the IEEE, 2015, pp. 1423-1440

[32] Gredler H, Medved J, Previdi S, Farrel A, Ray S, North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP, RFC 7752, 2016, https://tools.ietf.org/html/rfc7752

[33] Ash J, Le Roux J, Path Computation Element (PCE) Communication Protocol Generic Requirements RFC 4657, 2006, https://tools.ietf.org/html/rfc4657

[34] Rzym G, Wajda K, Chołda P, SDN-Based WAN Optimization: PCE Implementation in Multi-Domain MPLS Networks Supported By BGP-LS, Image Processing & Communications, 2017, pp. 35-48

[35] Rotsos C, King D, Farshad A, Bird J, Fawcett L, Georgalas N, Gunkel M, Shiomoto K, Wang A, Mauthe A, Race N, Hutchison D, Network service orchestration standardization: A technology survey, Computer Standards & Interfaces, 2017, pp. 203-215

[36] Molina E, Jacob E, Software-defined networking in cyber-physical systems: A survey, Computers and Electrical Engineering, 2018, pp. 407-419

[37] OpenFlow Management and Configuration Protocol, ONF Open Networking Foundation, 2014, https://www.opennetworking.org/wp-content/uploads/2017/07/of-config-1.2.pdf

[38] Devlic A, John W, Skoldstrom P, A Use-Case Based Analysis of Network Management Functions in the ONF SDN Model, Proceedings of the IEEE, 2012, pp. 85-90

[39] Emmerich P, Raumer D, Gallenmuller S, Wohlfart F, Carle G, Throughput and Latency of Virtual Switching with Open vSwitch: A Quantitative Analysis, Journal of Network and Systems Management, 2018, pp. 314-338

[40] Pfaff B, Davie B, The Open vSwitch Database Management Protocol, RFC 7047, 2013, https://tools.ietf.org/html/rfc7047

[41] Volpato F, Pereira Da Silva M, Goncalves A, Ribeiro M, An Autonomic QoS management architecture for Software-Defined Networking environments, Proceedings of the IEEE, 2017, pp. 418-423

[42] Schonwalder J, Overview of the 2002 IAB Network Management Workshop, RFC 3535, 2003, https://tools.ietf.org/html/rfc3535

[43] Schonwalder J, Bjorklund M, Shafer J, Network configuration management using NETCONF and YANG, IEEE Communications Magazine, 2010, pp. 166-173.